13.1 - Clustering | Unsupervised Learning

Input
- K (number of clusters)
- Training set $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$

$x^{(i)} \in \mathbb{R}^n$ (drop $x_0 = 1$ convention)

Randomly initialize K cluster centroids $\mu_1, \mu_2, \ldots, \mu_K \in \mathbb{R}^n$

Repeat {

Cluster assignment [for $i = 1$ to m
$c^{(i)} :=$ index (from 1 to K) of cluster centroid closest to $x^{(i)}$

Move centroid [for $k = 1$ to K
$\mu_k :=$ average (mean) of points assigned to cluster k

13.2 - Optimization Objective

$c^{(i)}$

μ_K

$\mu_{c^{(i)}} =$ cluster centroid of cluster to which example $x^{(i)}$ has been assigned

Optimization objective :

$$J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots \mu_K) = \frac{1}{m} \sum_{i=1}^{m} \| x^{(i)} - \mu_{c^{(i)}} \|^2$$

$$\min_{\substack{c^{(1)}, \ldots c^{(m)}, \\ \mu_1, \ldots \mu_K}} J(c^{(1)}, \ldots c^{(m)}, \mu_1, \ldots \mu_K)$$

13.3 - Random Initialization

Should have $K < m$

Randomly pick K training examples

Set $\mu_1, \mu_2 \ldots \mu_K$ equal to these K examples.

For $i = 1$ to 100 {

randomly initialize K-means.

Run K-means. Get $c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K$.

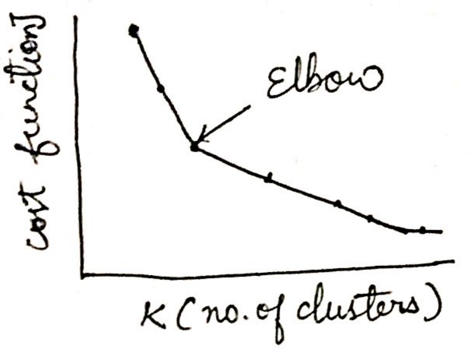
Compute cost function (distortion)

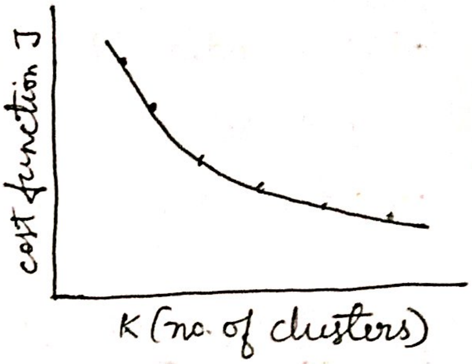
$J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K)$

Pick clustering that gave lowest cost $J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K)$

13.5 - Choosing the Number of Clusters

Elbow method:



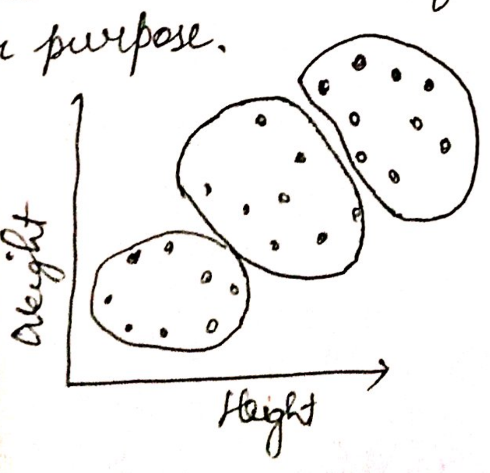


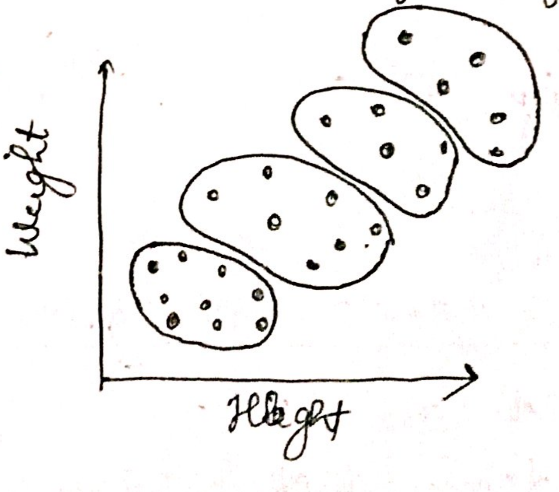
→ Elbow method is not always suitable because often we obtain the graph 2 with no distinct elbow.

Choosing the value of K

Sometimes, you running K-means to get clusters to use for some later / downstream purpose. Evaluate K-means based on a metric for how well it performs for that later purpose.

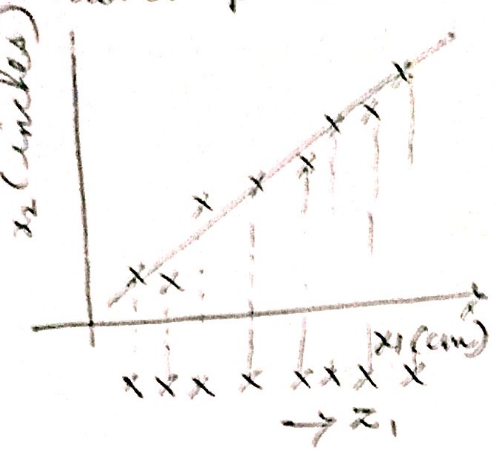
E.g.





14.1 – Dimensionality Reduction

Data compression



Reduction of data
2D to 1D

$x^{(1)} \in \mathbb{R}^2 \rightarrow z^{(1)} \in \mathbb{R}$

$x^{(2)} \in \mathbb{R}^2 \rightarrow z^{(2)} \in \mathbb{R}$

$\vdots$

$x^{(m)} \rightarrow z^{(m)}$

14.2 – Visualisation

→ Considering most important features
e.g. $\mathbb{R}^{50} \rightarrow \mathbb{R}^2$
Plotting it

14.3 – Principal Component Analysis

Principal Component Analysis (PCA) problem formulation

Reduce from 2-dimension to 1-dimension: Find a direction (a vector $u^{(1)} \in \mathbb{R}^n$) onto which to project the data so as to minimize the projection error.

Reduce from n-dimension to k-dimension: Find k vector $u^{(1)}, u^{(2)}, \ldots, u^{(k)}$ onto which to project the data, so as to minimize the projection error.

14.4 – Principal component analysis algorithm

Data preprocessing

Training set: $x^{(1)}, x^{(2)}, \ldots, x^{(m)}$

~~Preprocess~~ Preprocessing (feature scaling / mean normalization)

$$\mu_j = \frac{1}{m} \sum_{i=1}^{m} x_j^{(i)}$$

Replace each $x_j^{(i)}$ with $x_j - \mu_j$

If different features on different scales (e.g., x_1 = size of house, x_2 = no. of bedrooms), scale features to have comparable range of values

(PCA) algorithm

Reduce data from n-dimensions to k-dimensions

Compute "covariance matrix"

$$\Sigma = \frac{1}{m}\sum_{i=1}^{n} \underbrace{(x^{(i)})}_{n\times 1}\underbrace{(x^{(i)})^T}_{1\times n} \leftarrow n\times n$$

Compute "eigenvectors" of matrix Σ:

[U, S, V] = svd (Sigma); (code)

From [U, S, V] = svd (Sigma), we get

$$U = \left[\begin{array}{cccc} | & | & & | \\ u^{(1)} & u^{(2)} & \cdots & u^{(n)} \\ | & | & & | \end{array}\right] \in \mathbb{R}^{n\times n}$$

$$\underbrace{u^{(1)} \; u^{(2)}}_{k}$$

$$x \in \mathbb{R}^n \longrightarrow z \in \mathbb{R}^k$$

$$z = \underbrace{\left[\begin{array}{cccc} | & | & & | \\ u^{(1)} & u^{(2)} & \cdots & u^{(k)} \\ | & | & & | \end{array}\right]}_{\substack{n\times k \\ U_{reduce}}}^{T} x = \underbrace{\underbrace{\left[\begin{array}{c} \text{---}\; u^{(1)} \;\text{---} \\ \vdots \\ \text{---}\; u^{(k)} \;\text{---} \end{array}\right]}_{k\times n} \underset{\uparrow\, n\times 1}{x}}_{k\times 1}$$

→ After mean normalization (ensure features has zero mean) and optionally feature scaling:

$$\text{Sigma} = \frac{1}{m}\sum_{i=1}^{m}(x^{(i)})(x^{(i)})^T \qquad X = \left[\begin{array}{c} \text{---}\; x^{(1)T} \;\text{---} \\ \vdots \\ \text{---}\; x^{(m)T} \;\text{---} \end{array}\right]$$

$$\text{sigma} = \frac{1}{m} X' * X;$$

Code

```
[U S V] = svd (Sigma);
Ureduce = U(:, 1:k);
z = Ureduce' * x;
```

14.5 – Choosing the number of principal components

Average squared projection error: $\frac{1}{m}\sum_{i=1}^{m} \| x^{(i)} - x_{approx}^{(i)} \|^2$

Total variation in the data: $\frac{1}{m}\sum_{i=1}^{m} \| x^{(i)} \|^2$

Typically, choose k to be smallest value so that

$$\frac{\frac{1}{m}\sum_{i=1}^{m} \| x^{(i)} - x_{approx}^{(i)} \|^2}{\sum_{i=1}^{m} \| x^{(i)} \|^2} \leq 0.01 \quad (1\%)$$

99 % of variance is retained

<u>Algorithm</u>

Try PCA with $k = 1$

Compute $U_{reduce}, z^{(1)}, x^{(2)}, \ldots z^{(m)}, x_{approx}^{(1)}, \ldots x_{approx}^{(m)}$

Check if

$$\frac{\frac{1}{m}\sum_{i=1}^{m} \| x^{(i)} - x_{approx}^{(i)} \|^2}{\frac{1}{m}\sum_{i=1}^{m} \| x^{(i)} \|^2} \leq 0.01 \ ?$$

For a given k

$$1 - \frac{\sum_{i=1}^{k} S_{ii}}{\sum_{i=1}^{n} S_{ii}} \leq 0.01$$

→ $[U, S, V] = svd(Sigma)$

$$S = \begin{bmatrix} S_{11} & & & 0 \\ & S_{22} & & \\ & & S_{33} & \\ 0 & & & S_{nn} \end{bmatrix}.$$

$$\frac{\sum_{i=1}^{k} S_{ii}}{\sum_{i=1}^{n} S_{ii}} \geq 0.99$$

Choosing k (number of principal component)

Pick smallest value of k for which

$$\frac{\sum_{i=1}^{k} S_{ii}}{\sum_{i=1}^{m} S_{ii}} \geq 0.99$$

(99% of variance retained)

14.6- Reconstruction From Compressed Representation

$z = U^T_{reduce} x$

14.7- Dimensionality Reduction

supervised learning speedup

$(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})$

Extract inputs:

Unlabeled dataset : $x^{(1)}, x^{(2)}, \ldots x^{(m)} \in \mathbb{R}^{10000}$

PCA

$z^{(1)}, z^{(2)} \ldots z^{(m)} \in \mathbb{R}^{1000}$

New training set:

$(z^{(1)}, y^{(1)}), (z^{(2)}, y^{(2)}), \ldots (z^{(m)}, y^{(m)})$

<u>Note</u>: Mapping $x^{(i)} \rightarrow z^{(i)}$ should be defined by running PCA only on the training set. This mapping can be applied as well to the examples $x_{cv}^{(i)}$ and $x_{test}^{(i)}$ in the cross validation and test sets.

Application of PCA

- Compression
  - Reduce memory/disk needed to store data
  - Speed up learning algorithm
- Visualization
  - $k = 2$ or $k = 3$

<u>Bad use of PCA: To prevent overfitting</u>

Use $z^{(i)}$ instead of $x^{(i)}$ to reduce the number of features to $k < n$

Thus, fewer features less likely to overfit.

This might work OK, but isn't a good way to address overfitting. Use regularization instead

$$\min_{\theta} \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m} \sum_{j=1}^{n} \theta_j^2$$